Chapter 3

Data Type Definitions

Each PDS-archived product is described using label objects that provide information about the data types of stored values. The data elements DATA_TYPE, BIT_DATA_TYPE, and SAMPLE_TYPE appear together with related data elements that provide starting location and applicable length information for specific data fields. Within all PDS data object definitions, the byte, bit, and record positions are counted from left to right, or first to last encountered, beginning with 1.

Data values may be represented within data files as ASCII or BINARY format. The ASCII storage format is simpler to transfer between different hardware systems and often between different application programs on the same computer. However, strictly numeric data often are stored in binary numeric types, since the ASCII representation of most numeric values requires more storage space than does the binary format. For example, each 8-bit pixel value in an image file would require 3 bytes if stored in ASCII format.

3.1 Data Elements

Table 3.1 identifies the data elements that provide data type, location, and length information according to the objects in which they appear.

3.2 Data Types

Table 3.2 identifies the valid values that may appear for the DATA_TYPE, BIT_DATA_TYPE, and SAMPLE_TYPE data elements (or their aliases) in PDS data object definitions. Many of the values in this table have been aliased to other values. Providing aliases allows the PDS to support and maintain backward compatibility. However, the preferred method is to use the value rather than its alias.

Unless noted as ASCII, all values in the table are binary.

Table 3.1: Data-Type-Related Elements Used in Data Label Objects

Data Object	Data Elements	Notes
COLUMN	DATA_TYPE	
(without ITEMS)	START_BYTE	
	BYTES	
COLUMN	DATA_TYPE	ITEM_TYPE is an alias
(with ITEMS)	START_BYTE	
	BYTES (opt)	total bytes in COLUMN
	ITEMS	
	ITEM_BYTES	size for each ITEM
BIT_COLUMN	BIT_DATA_TYPE	
(without ITEMS)	START_BIT	
	BITS	
BIT_COLUMN	START_BIT	
(with ITEMS)	BITS (opt)	Total bits in BIT_COLUMN
	ITEMS	
	ITEM_BITS	size for each ITEM
IMAGE	SAMPLE_TYPE	
	SAMPLE_BITS	
HISTOGRAM	DATA_TYPE	ITEM_TYPE is alias
	BYTES (opt)	total bytes in HISTOGRAM
	ITEMS	
	ITEM_BYTES	size for each ITEM (bin)

Table 3.2: PDS Standard Data Types

Data Element Usage Codes:

D	=	DATA_TYPE
B	=	BIT_DATA_TYPE
S	=	SAMPLE_TYPE

Data Element Usage	**Value**	**Description**
D	ASCII_REAL	ASCII character string representation of real number
D	ASCII_INTEGER	ASCII character string representation of integer
D	ASCII_COMPLEX	ASCII character string representation of complex
D	BIT_STRING	alias for MSB_BIT_STRING
D, B	BOOLEAN	True/False indicator; 1, 2, or 4 byte unsigned number or 1-32 bit number; all 0's False; anything else True
D	CHARACTER	any ASCII character string
	COMPLEX	alias for IEEE_COMPLEX
D	DATE	ASCII character string representation of PDS date
D	EBCDIC_CHARACTER	any EBCDIC character string
	FLOAT	alias for IEEE_REAL
D	IBM_COMPLEX	IBM 360/370 mainframe complex number (8,16 byte)
D	IBM_INTEGER	IBM 360/370 mainframe 1, 2, and 4 byte numbers
D	IBM_REAL	IBM 360/370 mainframe real number (4 and 8 byte)
D	IBM_UNSIGNED_INTEGER	IBM 360/370 mainframe 1, 2, and 4 byte numbers
D	IEEE_COMPLEX	includes 8, 16, and 20 byte complex numbers
D, S	IEEE_REAL	includes 4, 8 and 10 byte real numbers
D	INTEGER	Single byte integers only
	INTEGER	alias for MSB_INTEGER (2+ bytes)
D	LSB_BIT_STRING	includes 1, 2, and 4 byte columns containing bit columns
D, S	LSB_INTEGER	includes 1, 2, and 4 byte numbers
D, S	LSB_UNSIGNED_INTEGER	includes 1, 2, and 4 byte numbers
	MAC_COMPLEX	alias for IEEE_COMPLEX
	MAC_INTEGER	alias for MSB_INTEGER
	MAC_REAL	alias for IEEE_REAL
	MAC_UNSIGNED_INTEGER	alias for MSB_UNSIGNED_INTEGER
D	MSB_BIT_STRING	includes 1, 2, and 4 byte columns containing bit columns

D, B	MSB_INTEGER	includes 1, 2, and 4 byte numbers
D, B, S	MSB_UNSIGNED_INTEGER	includes 1, 2, and 4 byte numbers, and 1-32 bit numbers
D, B	N/A	Used for spare (or unused) fields, if identified
D	PC_COMPLEX	includes 8, 16, 20 byte complex numbers
	PC_INTEGER	alias for LSB_INTEGER
D	PC_REAL	includes 4, 8, and 10 byte real numbers
	PC_UNSIGNED_INTEGER	alias for LSB_UNSIGNED_INTEGER
	REAL	alias for IEEE_REAL
	SUN_COMPLEX	alias for IEEE_COMPLEX
	SUN_INTEGER	alias for MSB_INTEGER
	SUN_REAL	alias for IEEE_REAL
	SUN_UNSIGNED_INTEGER	alias for MSB_UNSIGNED_INTEGER
D	TIME	ASCII character string representation of PDS date/time
	UNSIGNED_INTEGER	alias for MSB_UNSIGNED_INTEGER (2+bytes)
D, B, S	UNSIGNED_INTEGER	single byte numbers, or 1-32 bit numbers
	VAX_BIT_STRING	alias for LSB_BIT_STRING
D	VAX_COMPLEX	includes D, F, and H type complex numbers
	VAX_DOUBLE	alias for VAX_REAL
	VAX_INTEGER	alias for LSB_INTEGER
D, S	VAX_REAL	includes D (8 byte), F (4 byte), and H (16 byte) type real numbers
	VAX_UNSIGNED_INTEGER	alias for LSB_UNSIGNED_INTEGER
D	VAXG_COMPLEX	G type complex numbers only
D	VAXG_REAL	G type (8 byte) real numbers only

3.3 Binary Integers

There are two widely used formats for integer representations in 16-bit and 32-bit binary fields. These are the most-significant-byte first (MSB) and least-significant-byte first (LSB) architectures. The MSB architectures are used on IBM mainframes, many UNIX minicomputers (SUN, Apollo) and Macintosh computers. The LSB architectures are used on VAX systems and IBM PCs. The default interpretation for PDS labeled data is the MSB architecture, and non-specific data types (e.g. UNSIGNED_INTEGER) are aliased to MSB types. Therefore, files written on VAX or IBM PC hosts must specify LSB data types for binary integer fields, or use the appropriate aliases.

3.4 Signed versus Unsigned

The PDS default binary integer is a signed value in 2's complement notation. Therefore, a data type specified as INTEGER is interpreted as a signed integer. Unsigned binary integers must be identified using a valid UNSIGNED_INTEGER data type from Table 3.2.

3.5 Floating Point Formats

The PDS default representation for floating point numbers is in ANSI/IEEE standard. This representation is defined as the PDS IEEE_REAL data type, and aliases are identified in Table 3.2. Several specific floating point representations are supported by PDS, and are further described in Appendix C.

3.6 Bit String Data

A BIT_STRING data type is used for COLUMNs to hold individual bit field values. Each bit field is defined in a BIT_COLUMN object. A BIT_STRING data type can be a 1, 2, or 4 byte field, much like a binary integer. Extraction of specific bit fields within a 2 or 4 byte BIT_STRING is dependent on the host architecture (MSB or LSB), and follows the binary integer specifications identified in Section 3.3 above. In interpreting bit fields (BIT_COLUMNS) within a BIT_STRING, any necessary conversions (byte swapping from LSB to MSB) are done first, and then bit field (START_BIT, BITS) values are used to extract the appropriate bits. This will assure that bit fields are not fragmented due to differences in hardware architectures.

3.7 Format Specifications

The data format specification is used to determine the format for display of a data value.
The following FORTRAN data format specifications will be used:

Aw	Character data value.
Iw	Integer value.
Fw.d	Floating point value, displayed in decimal format.
Ew.d[Ee]	Floating point value, displayed in exponential format.

Where:

w=	Total number of positions in the output field (including sign, decimal point or “E”).
d=	Number of positions to the right of the decimal point.
e=	Number of positions in exponent length field.

3.8 Internal Representations of Data Types

Appendix C contains the detailed internal representation of the PDS standard data types listed in Table 3.2.

PDS has developed tools that are designed to use the specifications outlined in Appendix C for interpreting data values for display and validation.